Mellanox VMS Wizard 1.0.5

User Manual

Rev 1.0

NOTE:
THIS HARDWARE, SOFTWARE OR TEST SUITE PRODUCT (“PRODUCT(S)”) AND ITS RELATED DOCUMENTATION ARE PROVIDED BY MELLANOX TECHNOLOGIES “AS-IS” WITH ALL FAULTS OF ANY KIND AND SOLELY FOR THE PURPOSE OF AIDING THE CUSTOMER IN TESTING APPLICATIONS THAT USE THE PRODUCTS IN DESIGNATED SOLUTIONS. THE CUSTOMER'S MANUFACTURING TEST ENVIRONMENT HAS NOT MET THE STANDARDS SET BY MELLANOX TECHNOLOGIES TO FULLY QUALIFY THE PRODUCTO(S) AND/OR THE SYSTEM USING IT. THEREFORE, MELLANOX TECHNOLOGIES CANNOT AND DOES NOT GUARANTEE OR WARRANT THAT THE PRODUCTS WILL OPERATE WITH THE HIGHEST QUALITY. ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT ARE DISCLAIMED. IN NO EVENT SHALL MELLANOX BE LIABLE TO CUSTOMER OR ANY THIRD PARTIES FOR ANY DIRECT, INDIRECT, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES OF ANY KIND (INCLUDING, BUT NOT LIMITED TO, PAYMENT FOR PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY FROM THE USE OF THE PRODUCT(S) AND RELATED DOCUMENTATION EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Mellanox Technologies
350 Oakmead Parkway Suite 100
Sunnyvale, CA 94085
U.S.A.
www.mellanox.com
Tel: (408) 970-3400
Fax: (408) 970-3403

Mellanox Technologies, Ltd.
Beit Mellanox
PO Box 586 Yokneam 20692
Israel
www.mellanox.com
Tel: +972 (0)74 723 7200
Fax: +972 (0)4 959 3245

© Copyright 2014. Mellanox Technologies. All Rights Reserved.

Mellanox®, Mellanox logo, BridgeX®, ConnectX®, Connect-IB®, CORE-Direct®, InfiniBridge®, InfiniHost®, InfiniScale®, MetroX®, MLNX-OS®, PhyX®, ScalableHPC®, SwitchX®, UFM®, Virtual Protocol Interconnect® and Voltaire® are registered trademarks of Mellanox Technologies, Ltd.

ExtendX™, FabricIT™, Mellanox Open Ethernet™, Mellanox Virtual Modular Switch™, MetroDX™, TestX™, Unbreakable-Link™ are trademarks of Mellanox Technologies, Ltd.

All other trademarks are property of their respective owners.

Contents

Revision History **6**
Purpose **6**
1 Introduction **7**
2 VMS Configuration Overview **8**
3 MLAG Configuration Overview **9**
4 VMS Wizard Prerequisites **10**
4.1 Operating Systems 10
4.2 Python Packages 10
4.3 Puppet Server 10
4.4 Mellanox Switch Systems 10
5 Un/Installing the VMS Wizard **11**
6 CLI Tools **12**
6.1 High Level VMS Wizard Commands Flow 12
6.2 CLI Player 13
6.3 Scheme Planner 14
6.3.1 L3-VMS 14
6.3.2 L2-VMS 15
6.4 LLDP Discovery 17
6.5 VMS Scheme Diff 19
6.6 VMS Scheme Analyzer 20
6.6.1 L3-VMS 20
6.6.2 L2-VMS 20
6.7 VMS Configuration Planner 21
6.7.1 L3-VMS 21
6.7.2 L2-VMS 22
6.8 Converters 22
6.8.1 VMS CLI Converter 22
6.8.2 VMS Puppet Converter 22
6.9 MLAG Wizard 23
6.9.1 MLAG Tool Usage 23
6.9.2 Input File 23
6.9.3 Switches CSV 25
7 VMS GUI **26**
7.1 VMS Protocol Page 26
7.2 Scheme Planner Page 27
7.2.1 OSPF Protocol 27

7.2.2 MSTP Protocol 29
7.3 Switches Page 31
7.4 LLDP Discovery Page 32
7.5 Diff Visualization Page 33
7.6 Analyzer Page 34
7.7 Configuration Page 35
7.7.1 OSPF Protocol 35
7.7.2 MSTP Protocol 37

List of Figures

Figure 1 - VMS Planning Procedure ..... **Error! Bookmark not defined.**
Figure 2 - MLAG Planning Procedure 9
Figure 2 - VMS Protocol Page 26
Figure 4 - 3-Level OSPF Protocol Scheme Planner 27
Figure 5 - Action Log File 28
Figure 6 - MSTP Protocol Scheme Planner 29
Figure 7 - Action Log File 30
Figure 8 - Switches Page 31
Figure 9 - LLDP Discovery Page 32
Figure 10 - Diff Visualization Page 33
Figure 11 - Fat Tree Analyzer Page 34
Figure 12 - OSPF VMS Configuration Page 35
Figure 13 - MSTP VMS Configuration Page 37

Revision History

Table 1: Revision History

Rev	Date	Description
1.0	May 29, 2014	First release

Purpose

This manual describes how to use the VMS Wizard automated installation. The VMS Wizard is intended for server administrators and network administrators who are familiar with the Layer 2 and Layer 3 solutions for Mellanox Virtual Modular Switch™.

1 Introduction

The Mellanox Virtual Modular Switch™ (VMS) solution comprised of Mellanox 10GbE, 40GbE, and 56GbE fixed switches, provides an ideal and optimized approach for a fixed switch aggregation.

VMS is energy efficient and scales up to 28.8Tb/s of non-blocking bandwidth and up to 720 nodes of 40GbE and operates at ultra-low latencies.

The VMS can be set up in one of two modes – either in Layer 2 mode (L2-VMS) based on MSTP, or in Layer 3 mode (L3-VMS) based on OSPF.

VMS configuration and bring-up can be fully automated, from the early planning stages until it is operational, by leveraging the VMS Wizard. The VMS Wizard is a user interface application which operates a set of open-source command-line tools. The tools are modular and allow users step-by-step planning and configuration of the VMS solution. The VMS Wizard provides an automation environment to provision the fabric with a centralized application, an application that learns the way the switches interconnect and how they ought to operate in the data center.

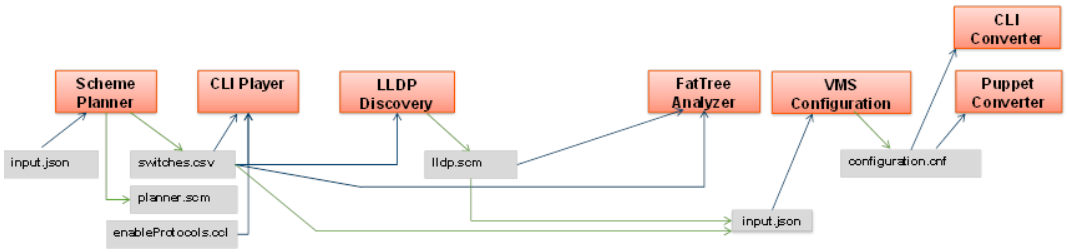
It starts with “paper design” prior to installing the switches. Once the fabric size is defined and the types of switches in the fabric are selected, the VMS Wizard specifies how to connect and configure the switches. After installation, the wizard verifies the connectivity and applies the configuration to the switches.

The wizard also provides automated configuration for two Mellanox Ethernet switches connected with an MLAG. These switches can be installed either at the edges of the VMS or in other network topologies.

2 VMS Configuration Overview

The flow of VMS planning is demonstrated in Figure 1. Each step has its required input and generates output which feeds into the stages that follow.

Figure 1 - VMS Planning Procedure



VMS configuration is composed of the following steps:

Step 1: Plan the fat tree deployment using the "Scheme Planner".
This step can be done prior to the actual installation.

Step 2: Enable required protocols on the switches using the "CLI Player".

Step 3: Discover existing wiring using "LLDP Discovery".

Step 4: Compare the existing fat tree to the planned scheme using the "Fat Tree Analyzer".

Step 5: Create a configuration scheme for the VMS using "VMS Configuration".

Step 6: Convert the configuration scheme to either CLI commands or Puppet manifests.

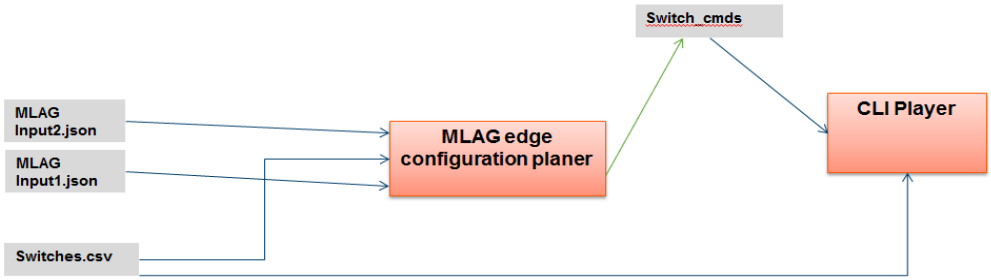
Step 7: Apply the scheme on the switches using "CLI Player" or Puppet.

The input and output of each step and tool are expanded upon further later on in this document.

3 MLAG Configuration Overview

The MLAG Edge Configurator is a standalone tool which offers a simple automation solution for configuring a pair of MLAG switches. The tool receives an input file and generates two CCL configuration files containing the CLI commands to configure the MLAG switch pair. The user can then use the CLI Player to configure the switches by running “switches.csv” and the folder containing the MLAG switches CCL files.

Figure 2 - MLAG Planning Procedure



Configuring MLAG is composed of the following steps:

Step 1: Prepare the input file.

Step 2: Execute “mlag_edge_configuration_planner –i <input file> -d <output folder>”.

Step 3: Repeat steps 1 and 2 for each MLAG pair.

Step 4: Prepare “switches.csv” with all the switches to be configured.

Step 5: Execute “vms_cli_player –c switches.csv –d <folder containing CCLs>”.

4 VMS Wizard Prerequisites

4.1 Operating Systems

- RedHat 6.X
- Centos 6.X

4.2 Python Packages

- python-2.6
- python-twisted-13.2
- python-netaddr
- net-snmp-python
- wxPython

4.3 Puppet Server

- Puppet Enterprise 2.7.1 (applicable when configuring VMS via Puppet).

4.4 Mellanox Switch Systems

To use the VMS tool, the following prerequisites are to be performed:

- Power up all switches
- Configure the management IP and the hostnames
- Verify that the installed management software is MLNX-OS version 3.3.5006 or later
- The switches should be configured only with management IPs and hostnames, other switch configuration may cause conflict with the VMS wizard

The tool supports the following switch systems: SX1036, SX1024, SX1012, SX1400, SX1700.

5 Un/Installing the VMS Wizard

➢ ***To install the VMS Wizard:***

1. Get the TAR file "vms-wizard" from the following URL http://www.mellanox.com/downloads/solutions/vms/. Run:

```
# wget
http://www.mellanox.com/downloads/solutions/vms/vms-wizard-1.0.5-2.el6.tar.gz
```

2. Unpack the TAR file using the CLI. Run:

```
# tar -xf vms-wizard-1.0.5-2.el6.tar.gz
```

3. Enter the VMS Wizard directory. Run:

```
# cd vms-wizard
```

4. Install the VMS Wizard. Run:

```
# ./vms_install.sh
Starting VMS Wizard Installation
…
…
…
Complete!
VMS Wizard Installation Completed.
VMS Wizard is installed on /opt/vms
#
```

The VMS Wizard is installed in the directory /opt/vms.

You also may custom install VMS Wizard using the following syntax:

./vms_install.sh [-o {vms_gui | vms_cli | cli_player}]

These parameters allow you to install a specific component of the VMS Wizard. You may install the VMS GUI (the default option when no parameter is specified), the CLI tools, or the CLI Player by using parameters vms_gui, vms_cli, or cli_player respectively.

➢ ***To uninstall the VMS Wizard:***

1. Initiate the uninstallation script. Run:

```
#/opt/vms/scripts/vms_uninstall.sh
```

2. Approve the uninstallation by typing "y" to the question.

```
Are you sure you want to remove VMS Wizard and it's dependencies? y
Uninstalling VMS Wizard...
```

6 CLI Tools

The VMS CLI tools are designed for Mellanox VMS bring-up and maintenance.

Table 2 - VMS CLI Tools Package

Tool	Description
CLI Player	Runs commands simultaneously on multiple switches. This tool is not only related to VMS.
Scheme Planner	Creates a connectivity map for the VMs installation prior to the installation process (off-site).
LLDP Discovery	Discovers the connectivity between the different VMS switches. The output of the tool is a LLDP discovery map.
VMS Scheme Diff	Creates a "diff" file between the scheme plan and the actual connectivity.
VMS Scheme Analyzer	Analyzes the output of the VMS scheme tool and verifies that the connectivity if according to the suggested plan.
VM Configuration Planner	Creates VMS topology configuration in REST format.
CLI/Puppet Converters	Convert from REST format to CLI commands or puppet manifests.
MLAG Edge Configurator	Creates an MLAG configuration f pair switches.

6.1 High Level VMS Wizard Commands Flow

- ***To configure a set of switches as VMS:***

1. Run Scheme Planner for your setup.

   For L3-VMS connectivity scheme:

   ```
   #/opt/vms/src/ospf/scheme_planner/vms_ospf_scheme_planner.py -i input.json
   ```

   For L2-VMS connectivity scheme:

   ```
   #/opt/vms/src/mstp/scheme_planner/vms_mstp_scheme_planner.py -i input.json
   ```

2. Update the file "switches.csv" with the switches' hostnames and credentials.
3. Run Scheme Planner again to update the switches' hostname in the connectivity scheme (planner.scm). Use the output (planner.scm) to connect your setup accordingly.

   For L3-VMS connectivity scheme, run:

   ```
   #/opt/vms/src/ospf/scheme_planner/vms_ospf_scheme_planner.py -i input.json
   -c switches.csv
   ```

   For L2-VMS connectivity scheme, run:

   ```
   #/opt/vms/src/mstp/scheme_planner/vms_mstp_scheme_planner.py -i input.json
   -c swtiches.csv
   ```

4. Enable LLDP and SNMP on all the VMS switches. Run:

   ```
   #/opt/vms/src/cli_player/vms_cli_player.py -c
   /opt/vms/examples/switches.csv -f
   /opt/vms/src/cli_player/config_templates/prereg.ccl
   ```

5. Run the LLDP discovery tool to get the physical connectivity scheme (lldplink.scm). Run:

```
#/opt/vms/src/snmp_lldp/vms_lldp_topology.py -i switches.csv
```

6. Run VMS Scheme Diff and VMS Scheme Analyzer to compare the network plan with the actual connectivity. Fix physical connectivity as needed. Run:

```
# /opt/vms/src/scheme_diff/vms_scheme_diff.py -p planner.scm -l lldplink.scm
- c switches.csv
```

For L3-VMS scheme analysis, run:

```
# /opt/vms/src/ospf/scheme_analyzer/vms_ospf_scheme_analyzer.py -l
lldplink.scm -p input.json -c switches.csv
```

For L2-VMS scheme analysis, run:

```
# /opt/vms/src/mstp/scheme_analyzer/vms_mstp_scheme_analyzer.py -l
lldplink.scm -p input.json -c switches.csv
```

7. Use Configuration Planner to generate a configuration plan (in REST format) for L2-VMS VMS or L3-VMS VMS.

For L3-VMS configuration plan, run:

```
#
/opt/vms/src/ospf/configuration_planner/vms_ospf_configuration_planner.py
-i config_input.json
```

For L2-VMS configuration plan, run:

```
#
/opt/vms/src/mstp/configuration_planner/vms_mstp_configuration_planner.py
-i config_input.json
```

8. Convert the plan to CLI commands or Puppet using the conversion tools.

For CLI commands, run:

```
# /opt/vms/src/cli_converter/vms_cli_converter.py -i configuration.cnf
```

For a Puppet manifest, run:

```
# /opt/vms/src/ puppet _converter /vms_puppet_converter.py -i
configuration.cnf
```

9. Run CLI Player to configure the VMS switches according to plan.

For CLI, run:

```
#/opt/vms/src/cli_player/vms_cli_player.py -d switches -c switches.csv
```

For puppet, add vms_menifest.pp to the site.pp in the Puppet master.

Make sure the switches are cleared of any networking configuration before attempting to exectue the VMS configuration.

6.2 CLI Player

This tool has the ability to run commands simultaneously on multiple switches.

```
# /opt/vms/src/cli_player/vms_cli_player.py --help
usage: vms_cli_player.py [-h] [-f SINGLE] -c CSV [-r ROLE] [-d MULTIPLE] [-v]
                         [-n] [-b BACKUP_DIR] [-m COMMAND][-t TIMEOUT]
```

Optional arguments:

```
  -h, --help            show this help message and exit
```

```
  -f SINGLE, --single SINGLE
                        cli commands file to execute on all switches in csv
  -c CSV, --csv CSV     csv contains the list of switches
  -r ROLE, --role ROLE  the role of the switches to execute the command
                        (optional)
  -d MULTIPLE, --multiple MULTIPLE
                        directory contains switches cli commands files
  -v, --verbose         increase verbosity level to debug
  -n, --no-backup       no running config backup
  -b BACKUP_DIR, --backup-dir BACKUP_DIR
                        directory contains the running config backup
  -m COMMAND, --command COMMAND
                        cli command to execute on all switches in csv
  -t TIMEOUT, --timeout TIMEOUT
                        timeout for command in seconds. default 60 seconds.
Example:
#/opt/vms/src/cli_player/vms_cli_player.py -d switches -c switches.csv
```

If you use the CLI Player for upgradeing/downgrading the switch image, the argument "-t" must be added with 900 seconds (i.e. "-t 900").

6.3 Scheme Planner

6.3.1 L3-VMS

The following command creates a recommended L3-VMS connectivity scheme:

```
#/opt/vms/src/ospf/scheme_planner/vms_ospf_scheme_planner.py -i input.json -
d <output_dir>
```

Table 3 - L3-VMS input.json File Parameters

Parameter	Description	Possible Values
spine_type	The spine switch type	SX1036; SX1012; SX1700
tor_type	The ToR switch type	SX1036; SX1024; SX1012; SX1700; SX1400
leaf_type	The leaf switch type	SX1036; SX1012; SX1700
num_of_uplink_from_tor	The number of leafs to which a ToR connects	
uplink_width_from_tor	The number of physical links from a ToR to a higher tier	
uplink_width_from_leaf	The number of physical links from a leaf to a higher tier (spine)	
min_uplink_port_from_tor	The interface from which to begin allocating interfaces for uplink connections to the higher tier	
min_uplink_port_from_leaf	The interface from which to begin allocating interfaces for uplink connections to the higher tier (spine).	

mlag	Defines whether MLAG is enabled on the VMS topology. Default: false	true/false
mlag_start_port	Defines the first IPL port.	Depends on the number of ports on the ToR.
mlag_width	Defines the number of ports in the IPL LAG.	

2-Level L3-VMS example:

```
{
"spine_type": "SX1036",
"tor_type": "SX1036",
"num_of_spines": 2,
"num_of_tors": 2,
"uplink_width_from_tor": 2,
"min_uplink_port_from_tor": 19
}
```

2-Level L3-VMS with MLAG example:

```
{
"spine_type": "SX1036",
"tor_type": "SX1036",
"num_of_spines": 2,
"num_of_tors": 2,
"uplink_width_from_tor": 2,
"min_uplink_port_from_tor": 19
"mlag": true
"mlag_start_port": 35
"mlag_width": 2
}
```

3-Level L3-VMS example:

```
{
"spine_type": "SX1036",
"tor_type": "SX1036",
"leaf_type": "SX1036",
"num_of_spines": 2,
"num_of_leafs": 4,
"num_of_tors": 6,
"uplink_width_from_tor": 2,
"num_of_uplink_from_tor": 2,
"min_uplink_port_from_tor": 19,
"uplink_width_from_leaf": 2,
"min_uplink_port_from_leaf": 19
}
```

```
# opt/vms/src/ospf/scheme_planner/vms_ospf_scheme_planner.py --help
usage: vms_ospf_scheme_planner.py [-h] -i INPUT [-c CSV] [-d OUTPUT_DIR]
```

Arguments:

```
  -h, --help            show this help message and exit
  -i INPUT, --input INPUT
                        vms scheme planner input file (json format)
  -c CSV, --csv CSV     user switches csv (optional)
  -d OUTPUT_DIR, --output-dir OUTPUT_DIR
                        output directory (optional)
```

6.3.2 L2-VMS

The following command creates a recommended L2-VMS connectivity scheme:

```
# opt/vms/src/mstp/scheme_planner/vms_mstp_scheme_planner.py -i input.json –
d <output_dir>
```

Table 4 - L2-VMS input.json File Parameters

Parameter	Description	Possible Values
spine_type	The spine switch type	SX1036; SX1012; SX1700
tor_type	The ToR switch type	SX1036; SX1024; SX1012; SX1400; ;SX1700
num_of_spines	Number of spines	1-18
num_of_tors	Number of ToRs	1-36
uplink_width_from_tor	The number of physical links from a ToR to a spine.	1-5
width_between_spines	The number of physical links from a ToR to a spine.	1-5
min_uplink_port_from_tor	The interface from which to begin allocating interfaces for uplink connections to the spines.	2-max

Example:

```
{
"spine_type": "SX1036",
"tor_type": "SX1036",
"num_of_spines": 2,
"num_of_tors": 2,
"uplink_width_from_tor": 1,
"width_between_spines": 1,
"min_uplink_port_from_tor": 19
}

# opt/vms/src/mstp/scheme_planner/vms_mstp_scheme_planner.py --help
usage: vms_mstp_scheme_planner.py [-h] -i INPUT [-c CSV] [-d OUTPUT_DIR]
```

Arguments:

```
  -h, --help            show this help message and exit
  -i INPUT, --input INPUT
                        vms scheme planner input file (json format)
  -c CSV, --csv CSV     user switches csv (optional)
  -d OUTPUT_DIR, --output-dir OUTPUT_DIR
                        output directory (optional)
```

- ***To execute the L2-VMS Planner scheme, edit the input.json file and run:***

```
# cd /opt/vms/examples
# /opt/vms/src/mstp/scheme_planner/vms_mstp_scheme_planner.py -i
mstp_planner_input.json
INFO - connectivity scheme generated successfully
#
```

- ***To view the L2-VMS Planner output scheme, open the output file and run:***

```
# cat planner.scm
{
    "Links": [
        {
            "d_port": 1,
            "d_switch": "SX1036_spine_0",
            "s_port": 19,
```

```
            "s_switch": "SX1036_tor_0"
        },
        {
            "d_port": 1,
            "d_switch": "SX1036_spine_1",
            "s_port": 20,
            "s_switch": "SX1036_tor_0"
        },
        {
            "d_port": 2,
            "d_switch": "SX1036_spine_0",
            "s_port": 19,
            "s_switch": "SX1036_tor_1"
        },
        {
            "d_port": 2,
            "d_switch": "SX1036_spine_1",
            "s_port": 20,
            "s_switch": "SX1036_tor_1"
        },
        {
            "d_port": 35,
            "d_switch": "SX1036_spine_1",
            "s_port": 35,
            "s_switch": "SX1036_spine_0"
        },
        {
            "d_port": 36,
            "d_switch": "SX1036_spine_0",
            "s_port": 36,
            "s_switch": "SX1036_spine_1"
        }
    ]
}
#
```

6.4 LLDP Discovery

Prior to executing LLDP discovery the VMS switches must be enabled with LLDP and SNMP.

If physical connectivity changes are made, please wait a few minutes before running the LLDP discovery script.

- ***To enable the LLDP and SNMP servers via script:***

Make sure switches.csv is updated with the VMS hostnames.

```
#/opt/vms/src/cli_player/vms_cli_player.py -c
/opt/vms/examples/switches.csv -f
/opt/vms/src/cli_player/config_templates/prereg.ccl
…
…
INFO - Configuration Done!
#
```

The following command creates a connectivity scheme of the current VMS topology.

```
# /opt/vms/src/snmp_lldp/vms_lldp_topology.py -i switches.csv -d <output_dir>
```

Parameters:

-i switches.csv – the CSV file detailing the VMS switches

-d <output_dir> – the directory under which to create output

```
# cat lldplink.scm
{
    "Links": [
        {
            "d_port": "2",
            "s_port": "19",
            "s_switch": "r-qa-sit-sx04",
            "d_switch": "r-qa-sit-sx05"
        },
        {
            "d_port": "2",
            "s_port": "20",
            "s_switch": "r-qa-sit-sx04",
            "d_switch": "r-qa-sit-sx06"
        },
        {
            "d_port": "1",
            "s_port": "19",
            "s_switch": "r-qa-sit-sx03",
            "d_switch": "r-qa-sit-sx05"
        },
        {
            "d_port": "1",
            "s_port": "20",
            "s_switch": "r-qa-sit-sx03",
            "d_switch": "r-qa-sit-sx06"
        },
        {
            "d_port": "35",
            "s_port": "35",
            "s_switch": "r-qa-sit-sx05",
            "d_switch": "r-qa-sit-sx06"
        },
        {
            "d_port": "36",
            "s_port": "36",
            "s_switch": "r-qa-sit-sx05",
            "d_switch": "r-qa-sit-sx06"
        }
    ]
}
```

```
# /opt/vms/src/snmp_lldp/vms_lldp_topology.py --help
usage: vms_lldp_topology.py [-h] [-i INPUTFILE] [-all] [-d OUTPUT_DIR]
```

Arguments:

```
  -h, --help            show this help message and exit
  -i INPUTFILE, --inputfile INPUTFILE
                        Input file in CSV format containing spines, leafs and
                        TORs (default: switches.csv)
  -all, --all           will go over all topology regradless the switches in
                        the csv
  -d OUTPUT_DIR, --output-dir OUTPUT_DIR
                        output directory (optional)
```

6.5 VMS Scheme Diff

The following command creates a diff of the recommended and current VMS topologies.

```
# /opt/vms/src/scheme_diff/vms_scheme_diff.py -p planner.scm -l lldplink.scm -
c switches.csv -d <output_dir>
```

Parameters:

-p – requires the recommended VMS topology scheme

-l – requires the current VMS topology scheme

-c – the CSV file detailing the VMS switches

-d <output_dir> – the directory under which to create output

Example:

```
# /opt/vms/src/scheme_diff/vms_scheme_diff.py -p planner.scm -l lldplink.scm -c
switches.csv
INFO - Command invoked: ['/opt/vms/src/scheme_diff/vms_scheme_diff.py', '-p',
'planner.scm', '-l', 'lldplink.scm', '-c', 'switches.csv']
INFO - Items that appear on lldplink.scm and not on planner.scm
INFO - Items that appear on planner.scm and not on lldplink.scm
```

Check the following output files now: lldplink_diff.json and planner_diff.json.

```
# cat planner_diff.json
{}

# cat planner_hostnames.scm
{
    "Links": [
        {
            "d_port": 1,
            "d_switch": "r-qa-sit-sx05",
            "s_port": 19,
            "s_switch": "r-qa-sit-sx03"
        },
        {
            "d_port": 1,
            "d_switch": "r-qa-sit-sx06",
            "s_port": 20,
            "s_switch": "r-qa-sit-sx03"
        },
        {
            "d_port": 2,
            "d_switch": "r-qa-sit-sx05",
            "s_port": 19,
            "s_switch": "r-qa-sit-sx04"
        },
        {
            "d_port": 2,
            "d_switch": "r-qa-sit-sx06",
            "s_port": 20,
            "s_switch": "r-qa-sit-sx04"
        },
        {
            "d_port": 35,
            "d_switch": "r-qa-sit-sx06",
            "s_port": 35,
            "s_switch": "r-qa-sit-sx05"
        },
        {
            "d_port": 36,
            "d_switch": "r-qa-sit-sx05",
            "s_port": 36,
```

```
            "s_switch": "r-qa-sit-sx06"
        }
    ]
}
```

```
# /opt/vms/src/scheme_diff/vms_scheme_diff.py --help
usage: vms_scheme_diff.py [-h] -p PLANNER_SCHEME -l LLDP_SCHEME -c CSV
                          [-d OUTPUT_DIR]
```

Arguments:

```
  -h, --help            show this help message and exit
  -p PLANNER_SCHEME, --planner-scheme PLANNER_SCHEME
                        scheme created by scheme planner
  -l LLDP_SCHEME, --lldp-scheme LLDP_SCHEME
                        scheme discovered by lldp
  -c CSV, --csv CSV     switches csv file (optional)
  -d OUTPUT_DIR, --output-dir OUTPUT_DIR
                        output directory (optional)
```

6.6 VMS Scheme Analyzer

This tool validates the connectivity scheme makes a viable VMS topology.

6.6.1 L3-VMS

```
#./vms_ospf_scheme_analyzer.py --help
usage: vms_ospf_scheme_analyzer.py [-h] -l LINKS_SCHEME -p PARAM_SCHEME -c CSV
```

Arguments:

```
  -h, --help            show this help message and exit
  -l LINKS_SCHEME, --links_scheme LINKS_SCHEME
                        vms scheme connectivity
  -p PARAM_SCHEME, --param_scheme PARAM_SCHEME
                        vms scheme parameters
  -c CSV, --csv CSV     switches csv file
```

6.6.2 L2-VMS

```
# /opt/vms/src/mstp/scheme_analyzer/vms_mstp_scheme_analyzer.py --help
usage: vms_mstp_scheme_analyzer.py [-h] -l LINKS_SCHEME -p PARAM_SCHEME -c CSV
```

Example:

```
# /opt/vms/src/mstp/scheme_analyzer/vms_mstp_scheme_analyzer.py -l
lldplink.scm -p mstp_planner_input.json -c switches.csv
INFO - Command invoked:
['/opt/vms/src/mstp/scheme_analyzer/vms_mstp_scheme_analyzer.py', '-l',
'lldplink.scm', '-p', 'mstp_planner_input.json', '-c', 'switches.csv']
INFO - No Duplicated Links Found!
INFO - No Links Between Same Tier Members Found!
INFO - Number of links in input JSON is equal to calculation
INFO - No invalid number of links active found for tor
INFO - No invalid number of links active found for spine
INFO - Done!
#
```

Arguments:

```
  -h, --help            show this help message and exit
  -l LINKS_SCHEME, --links_scheme LINKS_SCHEME
                        vms mstp scheme connectivity
  -p PARAM_SCHEME, --param_scheme PARAM_SCHEME
```

```
                        vms mstp scheme parameters
  -c CSV, --csv CSV     switches csv file
```

6.7 VMS Configuration Planner

6.7.1 L3-VMS

The L3-VMS Configuration Planner creates VMS topology configuration in REST form.

Parameters:

switches_file – lists all the switches in the VMS topology

connectivity_file – details the current VMS connectivity

min_vlan – defines the VLAN range allowed for the VMS configuration

max_vlan – defines the VLAN range allowed for the VMS configuration

ip_range – defines the IP range allowed for VMS configuration

switchport_mode – defines if the port is in “trunk” or “access” mode

use_lag – defines whether or not to use LAG

split_areas – adds the ability to configure multiple OSPF areas for the L3-VMS (3-level OSPF). The possible values are “true” or “false”. The default is “true”.

min_area_id – the minimum number of the area ID. If split_areas is configured to “false”, this parameter is the area ID. Otherwise, it becomes the minimum area number.

max_area_id – the maximum number of the area ID used if split-area is configured to “true”

mtu – the MTU of the port in bytes. Default is 1500. It will be configured on the physical port, as well as on the VLAN interface.

Example:

```
{
"switches_file":"switches.csv",
"connectivity_file":"lldplink.scm",
"min_vlan" : 10,
"max_vlan" : 20,
"ip_range" : "192.0.0.0/24",
"switchport_mode" : "trunk",
"use_lag" : false,
"split_areas": true,
"min_area_id": 1,
"max_area_id": 3,
"mtu": 1500
}
```

```
# ./vms_ospf_configuration_planner.py --help
usage: vms_ospf_configuration_planner.py [-h] -i INPUT [-d OUTPUT_DIR]

Optional arguments:
  -h, --help            show this help message and exit
  -i INPUT, --input INPUT
                        vms configuration input file (json format)
  -d OUTPUT DIR, --output-dir OUTPUT DIR
                        output directory (optional)
```

6.7.2 L2-VMS

This creates VMS topology configuration in REST form.

Parameters:

switches_file – lists all the switches in the VMS topology

connectivity_file – details the current VMS connectivity

min_vlan – the minimum VLAN ranges allowed for the VMS configuration

max_vlan – the maximum VLAN ranges allowed for the VMS configuration

consecutive – configures how to divide the VLANs between the roots of the MSTP. Possible values are: “true”; “false” (round-robin).

Example:

```
{
    "switches_file":"/opt/vms/examples/switches.csv",
    "connectivity_file":"/opt/vms/examples/lldplink.scm",
    "min_vlan" : 2,
    "max_vlan" : 100,
    "consecutive": false
}
```

```
# ./vms_mstp_configuration_planner.py  --help
usage: vms_mstp_configuration_planner.py [-h] -i INPUT [-d OUTPUT_DIR]

Optional arguments:
  -h, --help            show this help message and exit
  -i INPUT, --input INPUT
                        vms configuration input file (json format)
  -d OUTPUT_DIR, --output-dir OUTPUT_DIR
                        output directory (optional)
```

6.8 Converters

6.8.1 VMS CLI Converter

This tool takes the REST format configuration scheme and generates CLI commands for VMS configuration per switch.

```
# /opt/vms/src/cli_converter/vms_cli_converter.py --help
usage: vms_cli_converter.py [-h] -i INPUT [-d output_dir]

Optional arguments:
  -h, --help            show this help message and exit
  -i INPUT, --input INPUT
                        vms configuration input file (json format)
  -d output dir, --output-dir output dir
                        output dir for all switches cli commands
```

Example:

```
# /opt/vms/src/cli_converter/vms_cli_converter.py -i configuration.cnf
```

6.8.2 VMS Puppet Converter

This tool takes the REST format configuration scheme and generates a Puppet manifest for VMS configuration for all the switches.

```
# opt/vms/src/puppet_converter/vms_puppet_converter.py --help
usage: vms_puppet_converter.py [-h] -i INPUT [-d output_dir]
```

```
Optional arguments:
  -h, --help            show this help message and exit
  -i INPUT, --input INPUT
                        vms configuration input file (json format)
  -d output_dir, --output-dir output_dir
                        output dir for all switches cli commands
```

Puppet is not supported for L2 VMS (MSTP).

6.9 MLAG Wizard

6.9.1 MLAG Tool Usage

Run the following command to use the MLAG Edge Configurator.

```
./mlag_edge_configuration_planner  -i/input-file  -d/output-dir -h

-i/input-file - The input Json file contains the MLAG configuration.
-d/output-dir - The output folder to put the 2 switches ccl files.
-h/help - print this usage
```

The MLAG Wizard is a command line tool and does not have a GUI.

6.9.2 Input File

The .json input files should contain the following fields as described in the example below.

1. “switches” – dictionary containing the following:
   a. "hosts" – list containing 2 switches. Master switch and slave switch hostname/IP.
   b. “vip” – management virtual IP
   c. “subnet” – management subnet
   d. “switches_csv” – file contains all the switches see section 6.9.3
   e. “area_id” – user defined area ID. If not specified it is taken form CSV. If “none” no OSPF area is assigned.
2. “ipl” – dictionary containing the following fields:
   a. “min_port” – IPL minimum port
   b. “max_port” – IPL maximum port
   c. “vlan” – the IPL VLAN ID
   d. “network” – the network of the IPL (e.g. “192.168.0/30”)
   e. “lag_mode” – the mode the MLAG on the IPL. Supported values optional are “on”, “active”, or and “passive”. Default is “active”.
3. “mlag” – dictionary containing the following fields:
   a. “min_port” – MLAG minimum port
   b. “max_port” – MLAG maximum port

c. “width” – MLAG width

d. “mlag_group” – the MLAG group cluster

4. “mlag_switchport” – dictionary contains the following fields:

a. "mode" – MLAG switch_port mode [trunk/access]

b. “tagged_vlans” – MLAG tagged VLANs. Should be used only with trunk mode

c. “untagged_vlan” – MLAG untagged VLAN. Should be used only with access mode.

d. This dictionary may be empty.

5. “magp” – dictionary containing the following fields:

a. “vlan_to_subnet_mapping” – dictionary of VLAN-to-subnet mapping

b. ip_allocation – “low_to_high” or “high_to_low”. Default is “high_to_low”.

low_to_high – virtual IP X.X.X.1, left ToR X.X.X.2 right ToR X.X.X.3

high_to_low – virtual IP X.X.X.254, left ToR X.X.X.253, right ToR X.X.X.252

Example of the .json file:

```
{
    "switches": {
        "hosts": [
            "r-ufm-sw24",
            "r-ufm-sw53"
        ],
        "netmask": "255.255.252.0",
        "vip": "192.168.1.2",
        "switches_csv": "switches.csv",
        "area_id": 2
    },
    "ipl": {
        "min_port": "32",
        "max_port": "36",
        "vlan": "5",
        "network": "192.168.10.0/30",
        "lag_mode": "on"
    },
    "mlag": {
        "min_port": "5",
        "max_port": "10",
        "width": 2,
        "mlag_group": "mlag-group-1"
    },
    "mlag_switchport": {
        "mode": "trunk",
        "tagged_vlans": [ "10", "12-30", "40"]
    },
    "magp": {
        "vlan_to_subnet_mapping":{
         "2": "192.168.2.0/24",
         "3": "192.168.3.0/24",
         "11": "192.168.11.0/24"
        },
        "ip_allocation": "high_to_low"
   }
}
```

6.9.3 Switches CSV

The file “switches.csv” contains the switches in the MLAG setup. The role of these switches should be ToR.

The following is an example of a “switches.csv” file:

```
Role,Hostname,Username,Password,Community,Area
spine_0,r-ufm-sw42, admin, admin, public,
spine_1,r-ufm-sw54,admin,admin,public,
tor_0,r-ufm-sw55,admin,admin,public,
tor_1,r-ufm-sw52,admin,admin,public,
tor_2,r-ufm-sw51,admin,admin,public,
tor_3,r-ufm-sw50,admin,admin,public,
tor_4,r-ufm-sw53,admin,admin,public,4
tor_5,r-ufm-sw24,admin,admin,public,4
```

7 VMS GUI

The VMS GUI offers a convenient interface to run the VMS CLI tools.

The recommended resolution for using the VMS GUI is 1920x1080.

This section describes all the VMS GUI Wizard pages in detail.

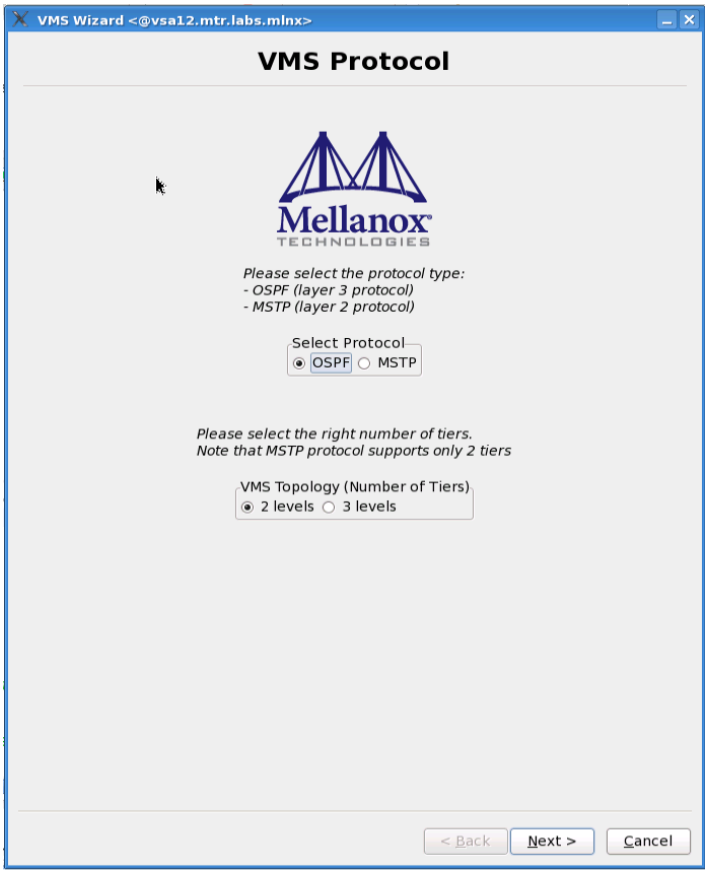
- ***To launch the VMS GUI, run:***

```
# /opt/vms/scripts/run_wizard.sh
```

7.1 VMS Protocol Page

This page allows you to select the protocols supported by the VMS wizard (OSPF or MSTP) and the number of tiers (2 or 3).

Figure 3 - VMS Protocol Page



What you select on this page affects the fields of the Scheme Planner page which comes afterwards.

MSTP protocol supports a 2-level tier only.

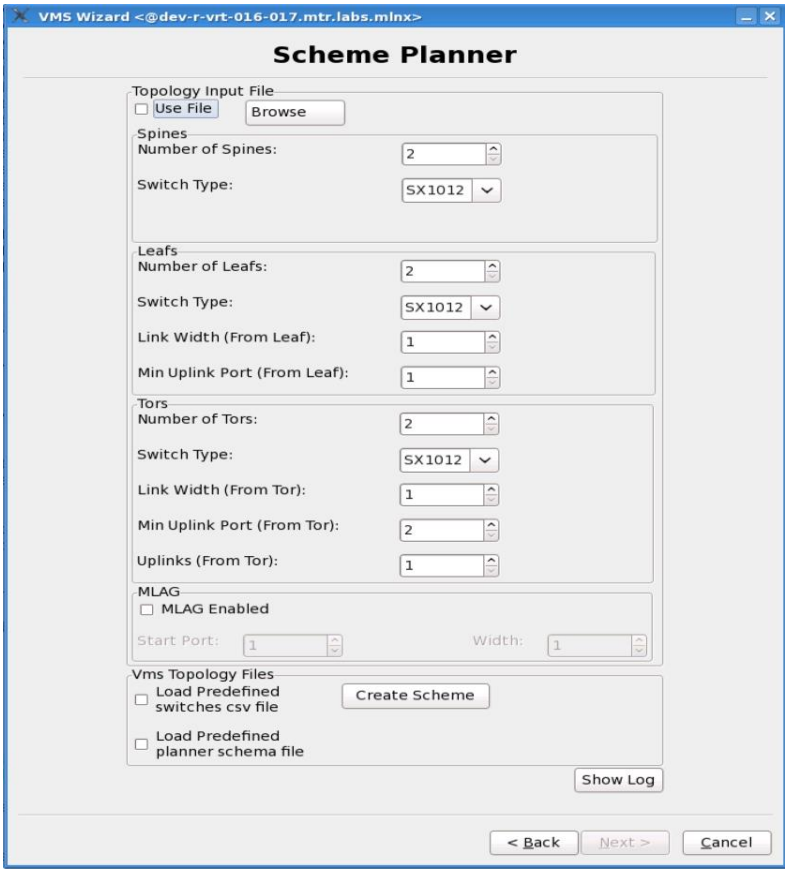
7.2 Scheme Planner Page

This page contains the input fields for the Scheme Planner.

7.2.1 OSPF Protocol

If OSPF is selected, depending on the selected number of tiers, the following page may appear:

Figure 4 - 3-Level OSPF Protocol Scheme Planner



For more details on the required fields and their definition, please refer to Section 6.3 L3-VMS on page 14.

This form may be completed in two methods.

Method I:

1. Fill the tiers data manually or upload a predefined input.json file with that information by checking the “Use File” box.

   If you are entering the data manually:

i. Fill the required fields regarding spine modules.

ii. Fill the required fields regarding leaf modules if relevant (3-layer L3-VMS only).

iii. Fill the required data regarding ToRs.

iv. Fill the required fields regarding the MLAG definitions.

2. (Optional) Load a predefined switches CSV file.
3. Click on the "Create Scheme" button to create a Scheme Planner file.

   If creating the scheme fails, you will be prompted and you must fix the errors in the input file and recreate the scheme to continue.
4. Click "Next".

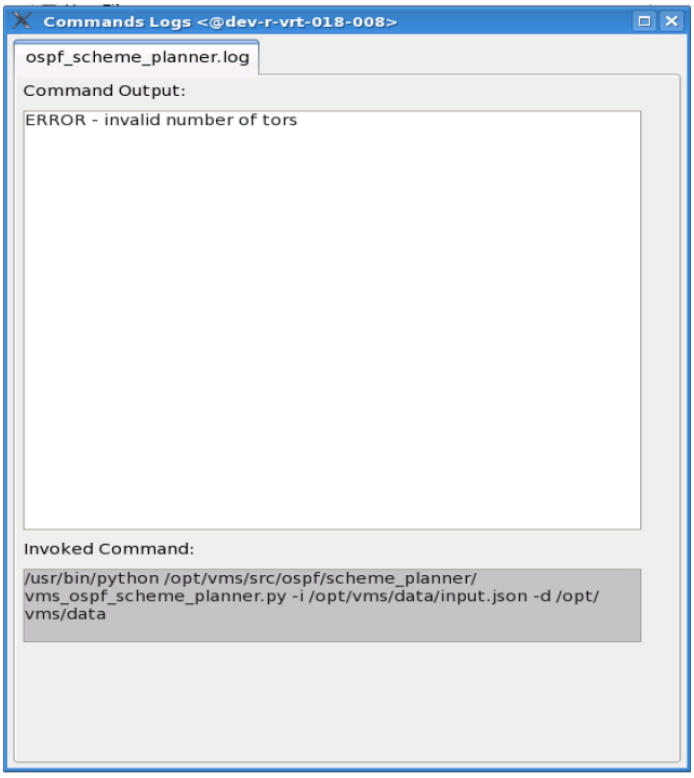
Method II:

1. Load a predefined Scheme Planner file by checking the "Load Predefined planner schema file" box.
2. Load a predefined switches CSV file by checking the "Load Predefined Switches csv file" box.
3. Click "Validate Scheme".

   If validation fails, you will be prompted and you must fix the errors in the input file and revalidate the scheme to continue.
4. Click "Next".

You may see the action log file by clicking "Show Log".

Figure 5 - Action Log File



Using the GUI runs the following CLI command: “opt/vms/src/ospf/scheme_planner /vms_ospf_scheme_planner.py -i input.json –d <output_dir>”.

For more details, refer to Chapter 6, “”.

7.2.2 MSTP Protocol

If MSTP is selected, the following page appears.

Figure 6 - MSTP Protocol Scheme Planner

For more details on the required fields and their definition, please refer to Section 6.3.26.3 L2-VMS on page 15.

This form may be completed in two methods.

Method I:

1. Fill the tiers data manually or upload a predefined input.json file with that information by checking the “Use File” box.

   If you are entering the data manually:

   i. Fill the required fields regarding spine modules.

ii. Fill the required data regarding ToRs.

2. (Optional) Load a predefined switches CSV file.
3. Click on the “Create Scheme” button to create a Scheme Planner file.

   If creating the scheme fails, you will be prompted and you must fix the errors in the input file and recreate the scheme to continue.

4. Click “Next”.

Method II:

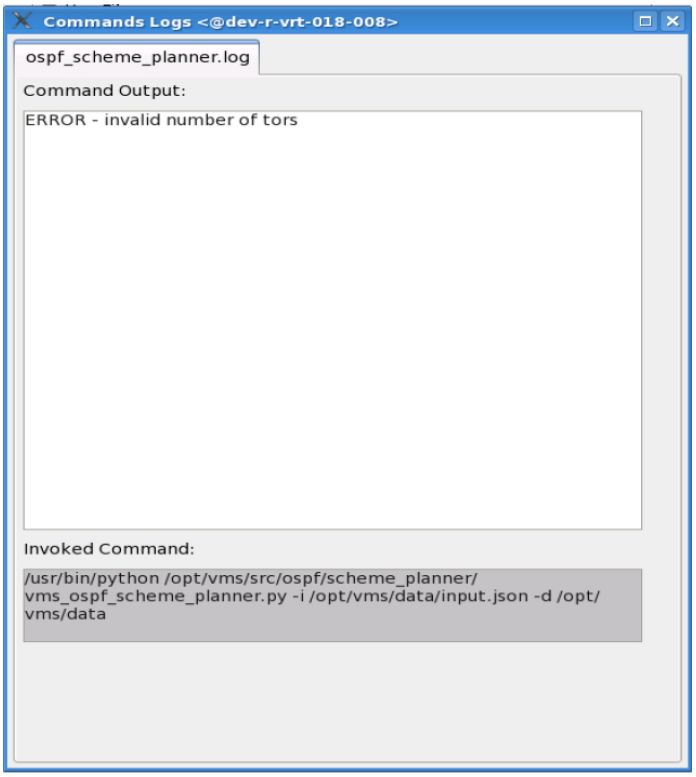
5. Load a predefined Scheme Planner file by checking the “Load Predefined planner schema file” box.
6. Load a predefined switches CSV file by checking the “Load Predefined Switches csv file” box.
7. Click “Validate Scheme”.

   If validation fails, you will be prompted and you must fix the errors in the input file and revalidate the scheme to continue.

8. Click “Next”.

You may see the action log file by clicking “Show Log”.

Figure 7 - Action Log File



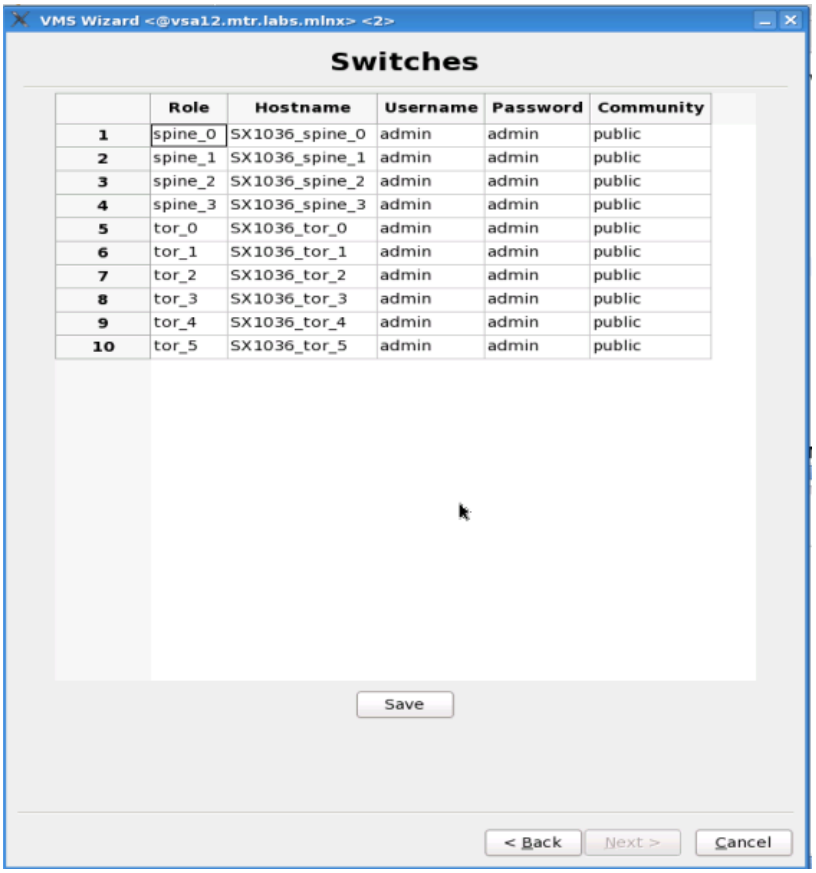
L2-VMS exposes a new field: Width Between Spines.

This field defines how many cables are required between spines. If the number is bigger than 1 the wizard, creates and configures LAGs between the spines.

7.3 Switches Page

This page contains all of the switches' information. If needed, the switches' hostname, user, password, and community may be edited.

Figure 8 - Switches Page

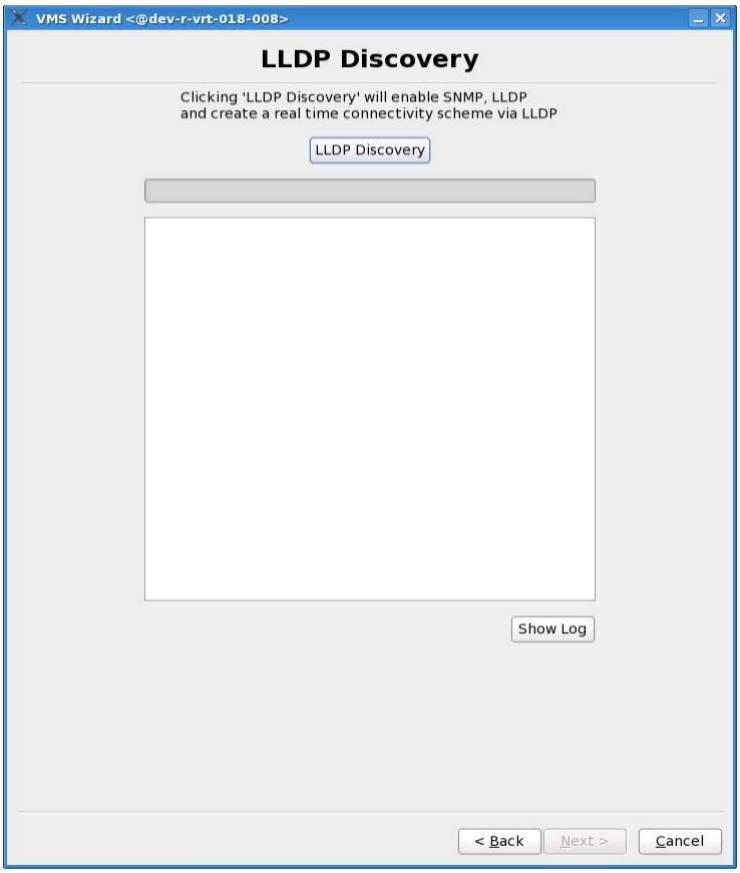


1. Insert a valid host name or IP address in the hostname column.
2. Edit the defaults for user, password, and community.
3. Save the VMS switches' data by clicking "Save".
4. Click "Next".

7.4 LLDP Discovery Page

This page enables SNMP and LLDP on the VMS switches and creates a discovery connectivity scheme via LLDP.

Figure 9 - LLDP Discovery Page



1. Click on “LLDP Discovery”.

   The discovery process could take a few minutes depending on how full the VMS is.

   The status of the discovery is presented in the main white panel.

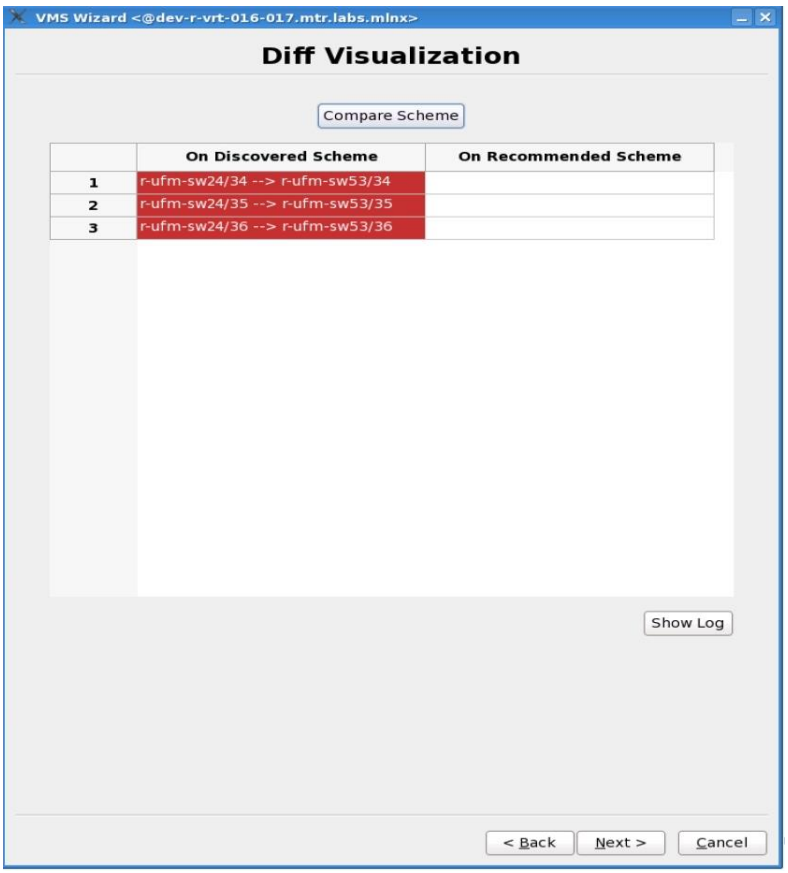
2. If errors occur, click “Show Log” to review the LLDP discovery log.
3. Click “Next”.

There are a few reasons that could lead to LLDP discovery errors. Please make sure to insert the correct hostnames, user, password, and community, and that ports are enabled prior to running LLDP discovery.

7.5 Diff Visualization Page

This page shows the diff between the LLDP connectivity scheme and the Scheme Planner scheme. In the left column appear the links which are discovered in the setup (the LLDP connectivity scheme) and do not appear in the planner scheme. The right column shows the opposite.

Figure 10 - Diff Visualization Page



1. Click “Compare Scheme”.
2. The diff between the schemes is presented in the main table.

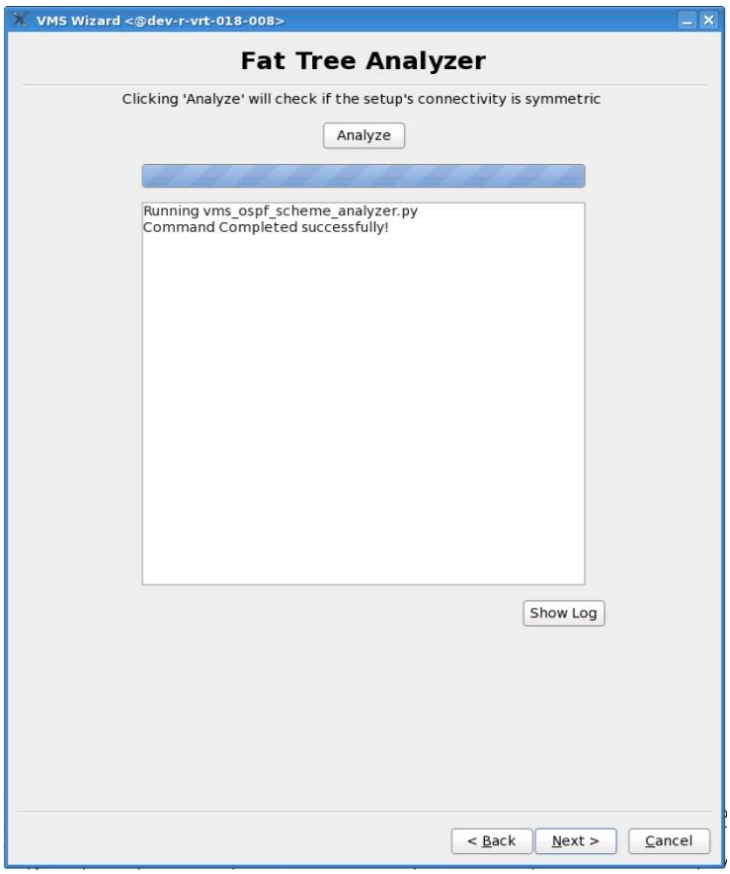
   If the table is empty click “Next”. If diffs appear, it is recommended to rearrange the lab cables as designed and repeat the last two pages (i.e. LLDP Discovery and Diff Visualization).
3. Click “Show Log” to see more information about the scheme differences.

To verify changes between other schemes click “Browse”.

7.6 Analyzer Page

This page analyzes the LLDP scheme and verifies the topology is a CLOS that consists of a fat tree (multi-root tree) of routing according to the “input.json” file.

Figure 11 - Fat Tree Analyzer Page



1. Click “Analyze”.
2. If errors appear on the main panel it means that the setup is not ready for configuration and it is recommended to rearrange the lab cabling according to the Scheme Planner scheme and go back to the “LLDP Discovery” page.
3. Click “Show Log” to see more information about action failure.
4. Click “Next”.

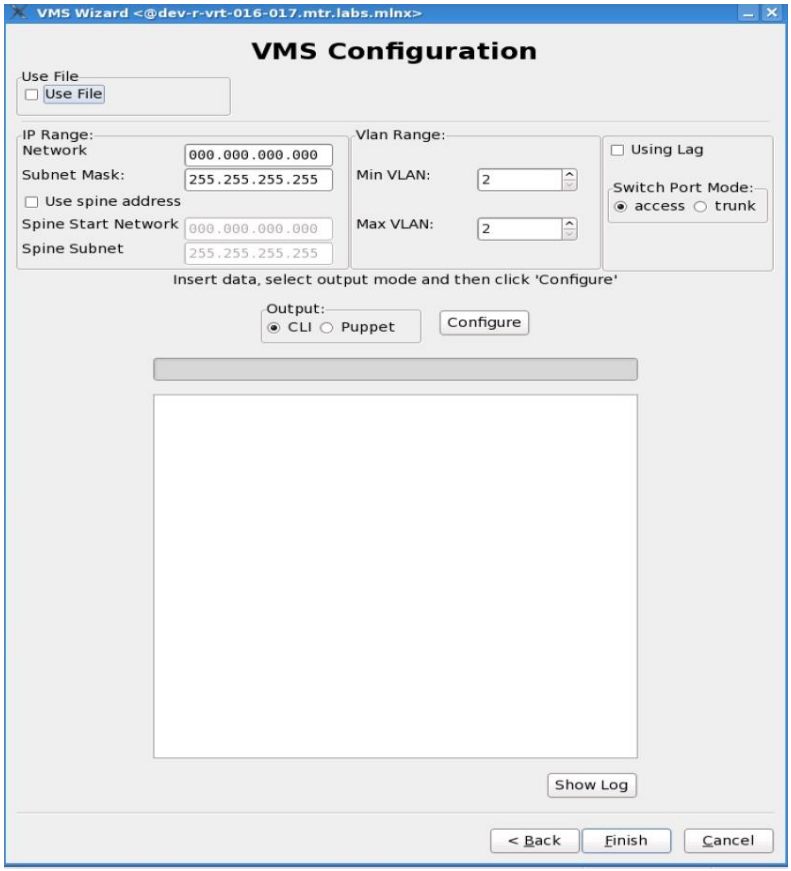
7.7 Configuration Page

This page contains the configuration parameters for the VMS.

7.7.1 OSPF Protocol

If OSPF is selected, the following page may appear:

Figure 12 - OSPF VMS Configuration Page



Using the GUI runs the following CLI command: "/opt/vms/src/ospf/configuration_planner/vms_ospf_configuration_planner.py -i config_input.json -d (--output-dir)". This creates a VMS topology configuration in REST form. This output may be used in the CLI or Puppet convertors (which yield CLI commands or a Puppet manifest to modify).

1. Fill the fields "Network" and "Subnet Mask".
2. Insert values for "Min VLAN" and "Max VLAN".
3. If using LAG, check "Using Lag".
4. Select the switch port mode: access or trunk.
5. Select output type: CLI or Puppet.
6. Click "Configure" to export configuration to the switches.
7. (Puppet only) Disable all the ports of the ToRs. Run:

```
# /opt/vms/src/cli_player/vms_cli_player.py -c
/opt/vms/examples/switches.csv -r tor -f
```

```
/opt/vms/src/cli_player/config_templates/<ToRs
switch_type>/disable_ports.ccl
```

8. (Puppet only) Copy vms_manifest.pp from the directory /opt/vms/data/ into the Puppet master at /etc/puppetlabs/puppet/manifests/, and insert the line "`import 'vms_manifest'`" to the file site.pp.

"Spine Start Network" and "Spine Subnet" are optional fields.

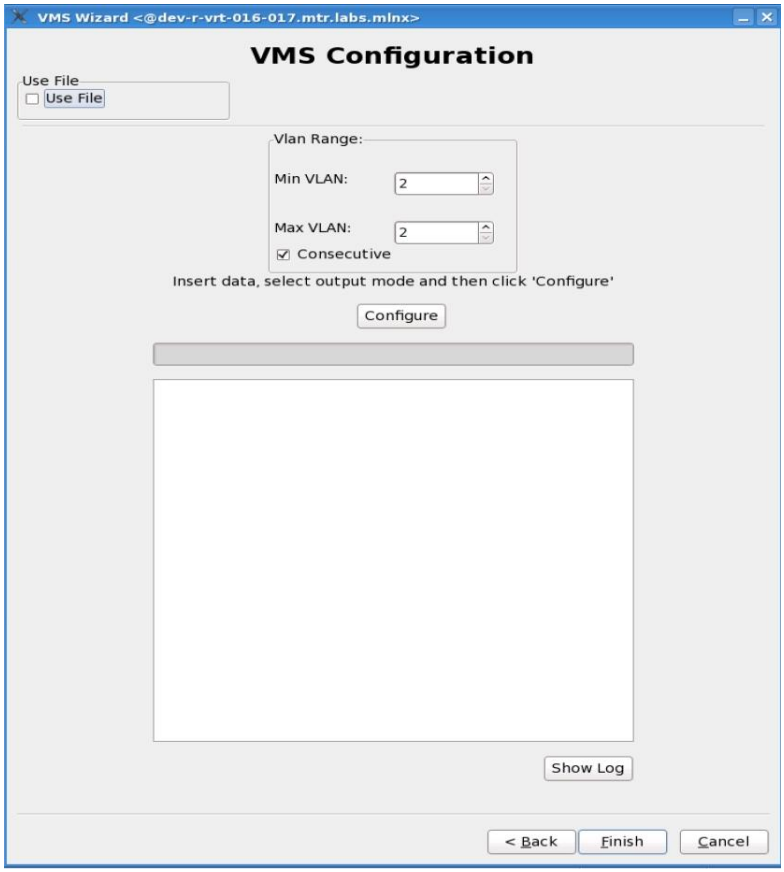
Before configuring the switches, the wizard shuts down all the interfaces. When the configuration done the wizard brings up all the interfaces.

The backup of the running config is saved in data/rc_backup.

7.7.2 MSTP Protocol

If MSTP protocol is selected, the following page may appear.

Figure 13 - MSTP VMS Configuration Page



1. Insert values for “Min VLAN” and “Max VLAN”.
2. Check “Consecutive” if you wish to have bulk of consecutive VLANs for each MSTP spine. Otherwise, the VLANs are assigned by way of a round-robin among the MSTP spines.
3. Click “Configure” to export configuration to the switches.

The backup of the running config is saved in data/rc_backup.